An Evolutionary Algorithm for Low Power VLSI Cell Placement

Mahmood R. Minhas
Information and Computer Science Department
King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia
E-mail: minhas@ccse.kfupm.edu.sa

Abstract - With the increasing use of battery operated mobile electronic devices, VLSI circuit designers are continuously focusing on approaches to low power designs. We present an evolutionary cell placement technique for low power VLSI standard cell placement. The proposed technique is based on two evolutionary algorithms namely Tabu Search and Genetic Algorithm. Experiments were carried out using representative circuits from ISCAS-85/89 benchmark suite. For the comparison purposes, we also implemented GA for our problem and compared placements results of the proposed technique to those of GA. The comparison shows that the proposed technique outperforms GA both in terms of quality of final placement solution obtained as well as CPU run time requirements.

I. Introduction

The need for low power driven VLSI design has emerged rapidly in past few years. While optimizing a circuit design for power consumption, other design objectives like performance and interconnect wire length need also to be taken care of. This fact leads to the development of techniques, which target to simultaneously optimize all these design goals. Previously, the objectives of optimizing interconnect wire length and performance were focused, and a large number of efforts targeting either one or both of above two objectives are reported in the literature [1], [2]. There has been reported some work for optimizing power consumption while considering the wire length and performance as constraints [3], [4]. Recently, some efforts targeting simultaneous optimization of all three objectives are also reported in [5], [6].

The organization of the rest of this paper is as follows: In the next section, we present the problem and formulate the cost functions. Section 3 presents the implementation details of our proposed approach. The experimental results and comparison are presented in Section 4.

II. VLSI Cell Placement and Cost Functions

VLSI design is a complex process and is carried out at certain abstraction levels [7]. The design process starts from an abstract idea, and then each intermediate step continues refining the design and the process ends with the fabrication of a new chip. The problem of power optimization can be addressed at a higher level as well as at a lower level e.g., physical level [8]. In this work, we address the above problem in the placement step at the physical level. Placement is an important step in VLSI physical design responsible for arrangement of cells on a layout surface for optimizing certain objectives while satisfying some constraints. Standard cell placement is a special case where all the cells to be placed have equal height.

Fig. 1. Various steps in VLSI design process.

We are addressing the problem at cell placement level with the objectives of optimizing power consumption, timing performance (delay), and wire length while considering layout width as a constraint. Formally, the problem can be stated as follows: A set of cells or modules \( M = \{m_1, m_2, ..., m_n\} \) and a set of signals \( S = \{s_1, s_2, ..., s_k\} \) is given. Moreover, a set of signals \( S_{m_i} \), where \( S_{m_i} \subseteq S \), is associated with each module \( m_i \in M \). Similarly, a set of modules \( M_{s_j} \), where \( M_{s_j} = \{m_i | s_j \in S_{m_i}\} \) is called a signal net, is associated with each signal \( s_j \in S \). Also, a set of locations \( L = \{L_1, L_2, ..., L_p\} \), where \( p \geq n \) is given. The problem is to assign each \( m_i \in M \) to a unique location \( L_j \), such that all of our objectives are optimized subject to our constraint [7].

A. Cost Functions

Now we formulate cost functions for our three said objectives and for the width constraint.

- **Wire length Cost:** Interconnect Wire length of each net in the circuit is estimated and then total wire length is computed by adding all these individual estimates:

\[
Cost_{wire} = \sum_{i \in M} l_i
\]

where \( l_i \) is the wire length estimation for net \( i \) and \( M \) denotes total number of nets in circuit (which is the same as number of modules for single output cells).
• **Power Cost:** Power consumption \( p_i \) of a net \( i \) in a circuit can be given as:

\[
 p_i \simeq \frac{1}{2} \cdot C_i \cdot V_{DD}^2 \cdot f \cdot S_i \cdot \beta \tag{2}
\]

where \( C_i \) is total capacitance of net \( i \), \( V_{DD} \) is the supply voltage, \( f \) is the clock frequency, \( S_i \) is the switching probability of net \( i \), and \( \beta \) is a technology dependent constant.

Assuming a fix supply voltage and clock frequency, the above equation reduces to the following:

\[
 p_i \simeq C_i \cdot S_i \tag{3}
\]

The capacitance \( C_i \) of cell \( i \) is given as:

\[
 C_i = C_i^r + \sum_{j \in M_i} C_i^g
\]

where \( C_i^g \) is the input capacitance of gate \( j \) and \( C_i^r \) is the interconnect capacitance at the output node of cell \( i \).

At the placement phase, only the interconnect capacitance \( C_i^r \) can be manipulated while \( C_i^g \) comes from the properties of the cell library used and is thus independent of placement. Moreover, \( C_i^r \) depends on wire length of net \( i \), so equation 3 can be written as:

\[
 p_i \simeq l_i \cdot S_i \tag{5}
\]

The cost function for total power consumption in the circuit can be given as:

\[
 Cost_{power} = \sum_{i \in M} p_i = \sum_{i \in M} (l_i \cdot S_i) \tag{6}
\]

• **Delay Cost:** Delay cost is determined by the delay along the longest path in a circuit. The delay \( T_{\pi} \) of a path \( \pi \) consisting of nets \( \{v_1, v_2, ..., v_k\} \), is expressed as:

\[
 T_{\pi} = \sum_{i=1}^{k-1} (C_{Di} + ID_i)
\]

where \( C_{Di} \) is the switching delay of the cell driving net \( v_i \) and \( ID_i \) is the interconnect delay of net \( v_i \). The placement phase affects \( ID_i \) because \( C_{Di} \) is technology dependent parameter and is independent of placement. Using the RC delay model, \( ID_i \) is given as:

\[
 ID_i = (L_{Fi} + R_i^t) \times C_i
\]

where \( L_{Fi} \) is load factor of the driving block, that is independent of layout, \( R_i^t \) is the interconnect resistance of net \( v_i \) and \( C_i \) is the load capacitance of cell \( i \) given in Equation 4.

The delay cost function can be written as:

\[
 Cost_{delay} = \max \{T_{\pi}\} \tag{9}
\]

• **Width Cost:** Width cost is given by the maximum of all the row widths in the layout. We have constrained layout width not to exceed a certain positive ratio \( \alpha \) to the average row width \( w_{avg} \), where \( w_{avg} \) is the minimum possible layout width obtained by dividing the total width of all the cells in the layout by the number of rows in the layout. Formally, we can express width constraint as below:

\[
 Width - w_{avg} \leq \alpha \times w_{avg} \tag{10}
\]

• **Overall Fuzzy Cost Function:** Since, we are optimizing three objectives simultaneously, we need to have a cost function that represents the effect of all three objectives in form of a single quantity. We propose the use of fuzzy logic to integrate these multiple, possibly conflicting objectives into a scalar cost function. Fuzzy logic allows us to describe the objectives in terms of linguistic variables. Then, fuzzy rules are used to find the overall cost of a placement solution. In this work, we have used following fuzzy rule:

**IF** a solution has

**SMALL wire length AND**

**LOW power consumption AND**

**SHORT delay**

**THEN** it is an **GOOD** solution.

![Fig. 2. Membership functions](image)

The above rule is translated to and-like OWA fuzzy operator [9] and the membership \( \mu(x) \) of a solution \( x \) in fuzzy set **GOOD solution** is given as:

\[
 \mu(x) = \begin{cases} 
 \beta \cdot \min_j \{\mu_j(x)\} + (1 - \beta) \cdot \frac{1}{3} \sum_j \mu_j(x); & \text{if } Width - \text{w}_{avg} \leq \alpha \cdot \text{w}_{avg}, \\
 0; & \text{otherwise}.
\end{cases} \tag{11}
\]

Here \( \mu_j(x) \) for \( j = p, d, l, width \) are the membership values in the fuzzy sets **LOW power consumption**, **SHORT delay**, and **SMALL wire length** respectively. \( \beta \) is the constant in the range \([0, 1]\). The solution that results in maximum value of \( \mu(x) \) is reported as the best solution found by the search heuristic.

The membership functions for fuzzy sets **LOW power consumption**, **SHORT delay**, and **SMALL wire length** are shown in Figure 2. We can vary the preference of an objective \( j \) in overall membership function by changing the value of \( g_j \). The lower bounds \( O_j \) for
different objectives are computed as given in Equations 12-15:
\[ O_l = \sum_{i=1}^{n} l_i^* \quad \forall v_i \in \{v_1, v_2, \ldots, v_n\} \quad (12) \]
\[ O_p = \sum_{i=1}^{n} Sd_i^* \quad \forall v_i \in \{v_1, v_2, \ldots, v_n\} \quad (13) \]
\[ O_d = \sum_{j=1}^{k} C D_j + ID_j^* \quad \forall v_j \in \{v_1, v_2, \ldots, v_k\} \text{ in path } \pi_c \quad (14) \]
\[ O_{\text{width}} = \frac{\sum_{i=1}^{n} \text{Width}_i}{\# \text{ of rows in layout}} \quad (15) \]
where \( O_j \) for \( j \in \{l, p, d, \text{width}\} \) are the optimal costs for wire-length, power, delay and layout width respectively, \( n \) is the number of nets in layout, \( l_i^* \) is the optimal wire-length of net \( v_i \), \( C D \) is the switching delay of the cell \( i \) driving net \( v_i \), \( ID \) is the optimal interconnect delay of net \( v_i \) calculated with the help of \( l_i \), \( S_i \) is the switching probability of net \( v_i \), \( \pi_c \) is the most critical path with respect to optimal interconnect delays, \( k \) is the number of nets in \( \pi_c \) and \( \text{Width}_i \) is the width of the individual cell driving net \( v_i \).

### III. The Proposed Technique and Implementation Details

In this section, we first present the proposed algorithm and then discuss its implementation details for low power VLSI placement.

An interesting novel idea is the introduction of a population of solutions instead of single solution in Tabu Search algorithm (TS) [10]. This is likely to enhance the power of TS by allowing it to visit the search space in a parallel fashion. The algorithm starts by taking a random initial population of solutions. Then, for each individual in the population, a certain number of neighbor solutions are generated and the best neighbor is found. A characteristic of this move is that not two dummy cells have the same value. To make it a perfect grid, we fill the empty locations by dummy cells represented by distinct negative integers as shown below. These negative numbers are used for encoding purpose as well as for the appropriate application of genetic operators like crossover and these do not play any role in cost computation of the solution.

| 3 | 5 | 8 | 6 |
| 9 | 10 |
| 7 | 11 | 1 |
| 4 | 2 |

In order to make it a perfect grid, we fill the empty locations by dummy cells represented by distinct negative integers as shown below. These negative numbers are used for encoding purpose as well as for the appropriate application of genetic operators like crossover and these do not play any role in cost computation of the solution.

| 3 | 5 | 8 | 6 |
| 9 | 10 | -1 | -2 |
| 7 | 11 | 1 | -3 |
| 4 | 2 | -4 | -5 |

In the initialization step random encoded strings are generated. For encoding purpose we use a square grid having \( L \) slots, such that \( L > N \), where \( N \) is the number of cells in the circuit. Each cell is assigned a positive integer value. Also, \( L - N \) dummy cells are created, each dummy cell is assigned a negative integer in such a way that not two dummy cells have the same value. To generate the string, first row of the grid is placed first in the string followed by the next row and so on.

### B. Cost Evaluation

Since, we are addressing a multiobjective optimization problem in which we are trying to minimize three mutually conflicting objectives, therefore we should have a measure which can quantify the overall quality of a solution with respect to all three objectives collectively. A conventional approach to this problem is the use of weighted sum. This approach is not used in our implementation because it is known to have certain problems. For instance, it is difficult to find values for weights as these heavily affect the relative importance of objectives.

In this approach, the costs of all the objectives are first fuzzy logic provides a convenient approach and hence used in this work. In this scheme, each solution is assigned a fitness value between 0 and 1 that is equal to the membership value of the fuzzy set of acceptable solution. This membership value is computed using Equation 11.

\[ \mu_{\text{accept}} = \frac{O_i}{O_i + O_p + O_d} \quad (16) \]

This membership value in the fuzzy set of acceptable solution.
IV. Experimental Results and Discussion

The proposed technique is applied for placement of some circuits from ISCAS-85/89 benchmark suites. For the comparison purposes, another well-known evolutionary algorithm was also implemented and applied to the above circuits. The performance of the proposed algorithm is compared with that of GA. The costs of the best solutions generated by both the approaches are listed in Table I. Here “L”, “P” and “D” represent the wire length, power and delay costs respectively, and “T” represents execution time in seconds. Layout width was constrained not to exceed more than 1.2 times the average row width by fixing the value of α in equation 10 equal to 0.2. This constraint is satisfied in obtaining all the results shown here.

The results of GA are obtained by best settings of its various algorithmic parameters. The parameters setting of the proposed technique for achieving these results is as follows. Total number of iterations run are 5000, which comprise of 2000 TS iterations and 3000 GA generations. The switch from TS to GA is made only once. The population size \( N_p \) used in TS part is 4 while in GA part the population size is 16 chromosomes. This fine tuning of parameters is made after careful study of the results obtained by choosing different settings. The population size in case of TS is reduced after observing that large population size increases run time of TS part without providing any significant performance. By adopting this measure, the run time requirement of the proposed technique are significantly lowered.

It can be observed from the results that in all of cases, the proposed technique produced solutions which are better in quality as compared to those obtained from GA. Also, the run time requirements of the proposed technique are considerably lower than that of GA.

V. Conclusions

We presented an evolutionary technique for low power VLSI cell placement. The proposed technique is applied for placement of some circuits from ISCAS-85/89 benchmark suites. The comparison shows the superiority of the proposed technique over GA in terms of placement quality and CPU run time requirements.

Acknowledgment:
Author thanks King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia, for all assistance and support provided for carrying out research.

References


